303Emulator

Marinus Kuivenhoven/Jeroen Schellekens

COLLABORATORS

	TITLE : 303Emulator		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Marinus Kuivenhoven/Jeroen Schellekens	December 25, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

303Emulator

1.1 Menu

```
                     303Emulator V2.2b  (updated june 1997)

    Choose the guide you want to read...


    Book 1:
                303emulator Manual V2.2b
                    Book 2:
                How to make 303scripts
                    Book 3:
                Other synth-projects
```

1.2 303Emulator

```
                   303Emulator V2.2b


                 About 303 Emulator

                 Requirements
                 and
                 Installation

                 A guide trough the program
                 or
                 Quick Reference

                 Examples

                 Distribution

                 History
                 and
                 Future
                 and
```

1.3 About 303 Emulator

303 Emulator V2

(c)1996-1997 by

Jeroen Schellekens
A Vintage Sound Render Program

What's the meaning of this program?
Well it generates samples sounding like the Roland TB-303, an analog box made in the early eighties. It produces a raw basssound, and is very useable for acid-style music (to say something). Many of you know what I'm talkinnabout so no words anymore about the silver box.

What I try to reach with this program(s) is to emulate the real thing. The problem is that you can't do this by calculating, or sampling the sound and then program a bassline or melody with it, because the kick comes when the sound gets sharper and duller while playing and altering the parameters, on a real 303 (or clone). So you can't just play one sample.

What's the solution?

-What most people do is sample a real 303 or any synth or cd and sample the sharpest and the dullest version of it. Then they emulate the cuttof frequency by fading between the sharp and the dull sample. THIS SOUNDS LIKE SHIT! Because the basics of sound don't work that way but with harmonics in related amplitudes related to the basic waveform. Every sound we hear is a package of movements of air, you can see them as sine-figures, and the frequency and volume of all those air-movements added in specific proportions we hear as one typical sound. I hope I explain it a bit well in my crappy englisch. When you fade from a sharp to a dull sound, and the sharp sound contains a sine at frequency 1000 Hz and the dull one contains a sine of 200 then the mix of the two samples in perfect balance would be a 50% volume of 1000 and a 50% volume of 200. But what should have been done is to get the middle of the two FREQUENCIES, so a new frequency of (1000+200)/2=600 should have been generated. Get it!?

If you want to know what the new plans are, click on

THIS BIG BUTTON!

1.4 Requirements

Before you start, make sure you have the following system:

Amiga with a harddrive with 2 Mb free space, 2 Mb memory (1 Mb Chip).
Prefered is a turboboard, altough an A1200-020 14 Mhz is fine too.

1.5 Installation

From now on (version 2.2, 2.2b and higher) an official install-program
is included with this package, so you can install the program on your HD
with this script.

If for some reason the installer does not work, you have to make a new
directory on your Harddisk with any name, like 'work:music'.
Then, copy everything from your original disk to this directory, the
program, the guide, the subdirs 'system', 'parameters', 'samples',
'scripts' and 'tempfiles'. Then, make in your user-startup an assign (do
this with a texteditor) like this: 'Assign 303emu2: work:music'. (Where
'work:music' depends on what your harddrive-name is and how you called your
just made subdir.)

1.6 Examples

All the programs include some examples, in different formats:

-Samples (8 bit Iff-format, can be loaded in many music-programs, these are
samples rendered in the 303emulator, and are single sounds but also loops)
Some are single sounds, and some are whole loops, generated with the
script-function

-Parameters (Settings of the Pots, Switches and Sliders, the parameters can
only be loaded in the 303emulator)

-Scripts (These scripts can only be loaded in the 303Emulator and the
scripteditor, and the last one generates and saves them also. Scripts
contain data about each single sample, in a list of more samples. In fact,
it are ASCII-textfiles, so you can edit them in a text-editor! To get a
list of commands available and some guidelines about affecting and
creating these scripts with a texteditor, read the '303script.help'-file)

-Textfiles and Guidefiles, wich include tips how to get acid-sounds, and
also info about how the 303-sound is build.

1.7 Usage

USAGE:

If you are curious about the 'Real'ness of the sound, wich is I think very
important, first listen to the prerendered samples, by loading them in a

sample-player or tracker-like-program. I hope you like them. Also, you can start the 303sequencer and load the example SampleBank. Then, load a Song and press 'Play Pattern' (For instructions about the 303sequencer, read the included docfile '303seq.help in the help-directory).
Now, while a melody is playing, drag the mouse to the left, and to the right. You hear something changing, don't you!? I hope you are impressed by the effect of it (And so not, I am sorry for you)!

-
Now, when you have heard and seen this, start the 303emulator. This is where all the sounds are generated. With each subject or parameter in the program I will describe, a simple drawing is included. For example, you see in the middle of the screen a black box. This is a display, that displays the current value of a rotating control, better called a potmeter. You can see a simple drawing of it in figure 4a1 as I told you.

```
 _________________
|                 |
| 0.96         1x |
|_________________|
```

(fig 4a1: the display in the middle of the screen)

The left value is that value of the potmeter, you are pointing on with your mouse. The right value is the setting of the step-factor (or a grid) when you change a potmeter by clicking on it with the left or right mousebutton. When you press the space-bar, this setting changes to a bigger value. If you keep pressing it, you see the first value is getting back. It's a sort of cycle-effect. The following settings are possible:

```
1x  - means increase or decrease a value with one
5x  - 5 times the effect of 1x
20x - 20 times the effect of 1x
```

-
Down the screen, you see a button called 'Render Sample' (fig 4a2).Push on it with your left mouse-button.

```
__________________________
|                         |
|          SAMPLE         |
|                         |
| Render   Save    Hear   |
|  ___      ___     ___   |
|  |_|      |_|     |_|   |
|_________________________|
```

(fig 4a2: buttons)

Now, a sample will be calculated. You see a graphic of the calculated piece. A sound exists out of a series of this pieces, joined after eachother. Such a piece they call a period. (fig 4b)

```
__________________________________________________________________________
|   /\                                                                   |
|  /  \                                                                  |
| /    \                                                                 |
|/     |         __                                                       |
|      |        /  \         _____                                        |
|      |       /    \       /     \         ______       __________________|
|       \     /      \     /       \       /      \_______/                 |
|        \   /        \___/        \______/                                |
|         \_/                                                              |
|                                                                         |
|                                                                         |
|_________________________________________________________________________|
```

(fig 4b: graphic of one period)

If it is ready, a short beep-sound will note you that it is. You are back in the program. Now, you can press a key on your keyboard, for example the 'w' to hear the rendered sound. To hear the same sample at a higher frequency, press on 'r'. You can use the number-keys and character-of-the-alphabet-keys (sorry, I'm not an englishman!) to play in the style of Sound- Pro- and so-on-trackers.

-
Why does the sample,you just rendered, sounds like it that?
Well, you see above in the screen a kind of box with 6 parameters (fig 4c).

```
/-----------------------------------------------------\
|[A]        First Sample Parameters (or single)       |
|                                                     |
| Tuning  CutOff  Resonance  Envelope  Decay  Accent |
|    _       _        _         _        _      _    |
|   /|\     /|\      /|\       /|\      /|\    /|\   |
|   \_/     \_/      \_/       \_/      \_/    \_/   |
\-----------------------------------------------------/
```

(fig 4c: Parameter-box with potmeters)

Every parameter or potmeter in this box has influence on the sound you render. Before I tell you about these parameters, read the following very well!

```
* Note: in the left corner of the parameter-boxes you see a flashing
* cursor around the [A] or [B]. If it flashes on the [A], this means
* that the settings of this box will be used. If you click on the [B] (fig
* 4d), then this set of parameters will be used for the rendering.
* This only counts for the 'single mode' (fig 4e)! So if you are in the
* sequence mode, the two parameter-boxes will function as sort of key-
* frames, where [A] is the first key-frame, and [B] the last one.
* The use of two different parameter-boxes in single mode, is that you can
* test the individual key frames (for later use in the sequence mode) by
* selecting the one you want to hear. (Updated may '97.)
```

```
/-----------------------------------------------------\
|[B]  Last Sample Parameters (only in Sequence mode) |
|                                                     |
| Tuning  CutOff  Resonance  Envelope  Decay  Accent |
|    _       _        _         _        _      _    |
|   /|\     /|\      /|\       /|\      /|\    /|\   |
|   \_/     \_/      \_/       \_/      \_/    \_/   |
\----------------------------------------------------/

(fig 4d: Second parameterbox with same potmeters)

 _____________
|             |
|   Single    |
|   _         |
|   \__      |
|   |\ |     |
|   |___|     |
|             |
|  Sequence   |
|_____________|

(fig 4e: Single & Sequence Mode)


Effect of the 6 potmeters (from the first and second parameter-box):

Tuning - determines the period of the sound. A higher value will cause a
         a lower note!!! This because the bigger the period, the longer it
         will take to hear the next one, so the sound gets lower. (A period
         is one wave of a sound) So it is a little confusing in the
         beginning (I think I change it for the next release).
         If you want a low sound, try a Tuning (period so) of 360. If you
         want an octave higher, you'll have to enter 180. This is a normal
         tone, when played on a frequency of \ensuremath{\pm} 20 kHz (key '3' on ↩
            the
         keyboard).

CutOff - sets the cutoff frequency of the filter that influences the sound.
         The lower the value of it, the lower it starts to cut of the
         higher frequencies of the sound. This type of filter is called a
         lowpass-filter. It passes the low frequences and filters out the
         frequences (drop them) that are higher than the value of the cut-
         off-parameter. So a high value means a crispy sound, and a low
         value means a dark, dull sound.
         A value of 10 to 20 has a very dull and bassy sound. Between 40
         and 80 it's a bit normal, and values of 90 to 130 get very
         tjilping and chirping. When you pass the border of 150 and higher,
         you get very electro-static sounds (because of aliasing effect,
         but its effect is nice!).

Resonance - sets the volume of the extra harmonic that comes with the
            lowpass-filter. The frequency (or pitch) of it is determined
            by the cuttof-frequency. Namely, the resonance is a peak-
            frequency precisely at the same frequency as the cuttof-point.
            The form of this resonance-sound is a sine. With the Resonance-
            parameter, you change the volume of this sine.
            This parameter is tricky (I'm working on it to get it better
            gripped) A value of 0 means there is NO resonance. A value of 1
            means there is MAXIMUM resonance. BUT, If you think that a
            value of 0.5 is half the resonance effect, you're WRONG! Check
            the following table and you can guess how it works in a way:
```

Reso-value	Description
0	No resonance
0->0.9	No resonance hearable, but volume of oscillator gets lower (at the moment more a Korg-filter than a Roland-filter, but will be changed in future.)
0.91	Very little resonance
0.96	Normal resonance
0.97	
0.98	Very many resonance
1	Extreme resonance

*** Values with 3 floats are possible, like 0.965 ***

Envelope - determines the drop down level and speed of the filter-frequency (the cut off-frequency). The higher the value, the higher the drop-down. Note: it’s a very sensitive parameter. A value of 0 means no decrease of the cuttof-freq. and a value of 1 means that in 2 periods the cutoff-freq reaches the lowest frequency possible! So here’s again a table:

Envelope-value	Description
0	No freq-decrease
0.01	Slight drop-down
0.2	Normal drop down
0.3	Heavy drop-down
0.5	Enormous decrease of cuttof-freq.

Decay - sets the length of the sound. You can only enter integer values. The value of the decay (scaled to a period of 360!) represents the number of periods. So if you enter a value of 18, The number of periods is, if the Tuning is 360, also 18. But if the same decay-value is used and the Tuning (thus the period) is 180, the finally rendered number of periods is (360/180)*18=36 periods! Why, you think? Well, it means that a note with a high pitch (thus a low period) has the same length as a low one, without affecting the decay-button.
Try to experiment with decay-values such as 20 and 30. This is not too long, so it renders quick, and it is long enough for you, when you play at freqs. of 18 kHz. (Often with TB-303_like sounds, you play short notes staccato)

Accent - gives the sound more aggressiveness. A value of 1 (lowest) means NO accent, and the maximum of 2 means MAXimum accent. It gives the sound more reso, more drop-down of the filter-freq. and mor volume. If you use it, I think you’d better set it to the MAX, or else totally not.

-
Now we’ve had the description of the two (they are identical) parameter-boxes, I’ll tell you about the params and switchers, at the top right. First you see the switch [Single/Sequence], (fig. 4e) Where you can switch between rendering 1 sample or a bunch of it.

If it directs to SINGLE, only the first_parameter_box (A) of the two ones has effect on it (already explained above). 'But what if I want to hear the second_parameter_box-settings?' Well, when you get one of the boxes by his head (like a Workbench-window), you can drag it, while holding a mouse-button. If you drag it to the other (place OVER it) an operation happens, when you loose the mousebutton. The LEFT-mousebutton REPLACES the old values with the holded ones, and the RIGHT-mousebuttons SWAPS the two parameter-boxes its values. So if you want to hear box (B) you get it with the right-mouse-button and lay it over box (A).

If the switch directs to SEQUENCE, and you click on RENDER SEQUENCE (fig. 4f) a FILE-REQUESTER will be opened. Then, you must give a basename of the samples that are going to be rendered (eg. '303sound.). The program renders the sounds after clicking [OK] and places its number after the filename ('303sound.001). If you click in the FILE-REQUESTER on [Cancel], you cancelled the whole rendering-process and end up in the main-prog again.

```
 ______________________
|                      |
|       Sequence       |
|                      |
|  Render     Samples  |
|   ____         _     |
|  |    |       /|\    |
|  |____|       \_/    |
|                      |
|______________________|
```

(fig 4f: sequence render options)

The number of samples to be rendered in this mode, you can adjust with the 'Samples'-potmeter (fig 4f). If you change it, the value will change, as you can see in the display in the middle. No tricks and jokes here: the number displayed represents the number of samples (default 32).

What the Sequence-rendering process does, is rendering the inbetween-settings between Parameter-box A (fig 4c) and Parameter-box B (fig 4d). Parameter-box A are settings for the first sample, Parameter-box B for the last, and everything between that the program calculates. It's a sort of key-framing effect. For example, if you have set the number of samples to be rendered is 8 (set with potmeter in fig 4f), each sample will have these kind of settings:

Sample	Tuning	Cutoff	Reson.... (and so on)	
1 (First Key)	200	80	0.99	<-----These are set by you
2	200	74.3	0.98	
3	200	68.6	0.97	
4	200	62.9	0.96	
5	200	57.1	0.95	
6	200	51.4	0.94	
7	200	45.7	0.93	
8 (Last Key)	200	40	0.92	<-----These are set by you

-

Then we get to right top with the switch [SAWTOOTH/SQUARE]. It's obvious that it changes the oscillator (the soundsource) between a square-waveform and a sawtooth-waveform. The trick of the square, as used on the original TB-303, is that (I think but am not sure!) it is a combination of two sawteeth at double frequency, and the second on reversed vertical and placed after the first one. Look at figure 4g.

```
Sawtooth:                     Square:

      |\                         |
      | \                        |\
      |  \                       | \
______|   \______             ___|  \__   ___
                                      |  /
                                      | /
                                      |/
                                      |
```

(fig 4g: graphic of a sawtooth and a square)

-
Glide-parameters.

A glide makes a sound glide, change in frequency, to another pitch. On a real TB-303, the glide- or portamento-effect lets note 1 glide to note 2, while note 1 is playing. In the 303emulator, it works a little the same, but also different. Because in the older version of this program I had implemented a stupid way of the glide-function, I rewrote it, but it had to be somewhat compatible with the old routine. That’s why it works not exactly the same as the original, altough the effect is the same. Look at the figure (fig 4h) and you see 4 thingies: 3 potmeters and a switch. The switch lets you decide wheter the note glides UP or DOWN. The problem of the glide-functions is, that you can’t give an end-note where the sound glides to. Instead of it you can set the LEVEL of the glide, so the frequency-change (per period). Then you set the START-period, so when it has to start gliding in the sound. You can set the stop-period with the STOP-meter.

```
 __________________________________
|                                  |
|              GLIDE               |
|     Level               Up       |
|       _                _         |
|      /|\               \__      |
|      \_/               |\ |     |
|                        |____|    |
|                                  |
|                        Down      |
|                                  |
|     Start              Stop      |
|       _                  _       |
|      /|\                /|\      |
|      \_/                \_/      |
|__________________________________|
```

(fig 4h: the glide-parameters.)

```
-
The monitor or tv-screen down-right. This monitor gives information of what
you are doing with some functions (Sorry 4 bad english). For example, it
can show the sample while rendering, but also gives some (error-)messages.

-
With the parameters-buttons (figure 4j) you can load and save all the
settings you've made in the program, including the settings of the plug-ins
(explained later) and the prefs-screen (also explained later).
However it does not save or load a rendered sound, or a script. The
parameters can only be loaded in this program and are also only made by
this program. You can't do anything else with it in contrast with a sample.

 _________________________
|                         |
|       Parameters        |
|  Load            Save   |
|   ___             ___   |
|  |   |           |   |  |
|  |___|           |___|  |
|_________________________|

(fig 4j: load and save parameters: the values of all switches & potmeters)

-
The Volume options. There are two potmeters for it (see fig 4k); one called
CLIP, wich boosts the sample until it gets to the clipping-level, this
means that the sound can't be registred harder, so a treshold or a border
cuts the waves that come over that border, with the effect of square-like
level-downs. It sounds more aggressive. (In fact it only boosts the sound
that comes beneath the 0-line, because above that 0-line the sound is often
maximum.) The second is distortion. But it's not kind-of guitar-distortion!
+-
| (If you truly want guitar-like distortion, then go to one of the plug-ins
| by clicking on the plug-in button middle left and search for 'OVERDRIVE'.
| But all this is explained later, so just read on and let it rest!)
|
+--> It's more a sort of sub-dis-harmonic that is synced with each period
     of sound, but its form changes every period. The effect is a low
     growling effect, wich fattens the sound (Nice for Moog-like sound!).
     It sounds the best with the sawtooh.

 ______________________
|                      |
|        Volume        |
|                      |
|  Clip       Distort  |
|    _           _     |
|   /|\         /|\    |
|   \_/         \_/    |
|______________________|

(fig 4k: Volume-controls)

-
Now let me explain the middle of the screen, with left a PREFS-button,
```

aside it a PLUG-IN-button, and then 6 small buttons from_left_to_right:
QUIT, UNDO, NEW, HELP, VIEW, SCRIPT. See fig 4m.

PREFS-button: Once you click on this boy, a new screen appears, layed over
the main-screen. This prefs-screen (fig 4l) contains some parameters (they
also are saved and loaded with LOAD & SAVE PARAMETERS (fig 4j)) wich you
will not use very often (I hope).

```
 _________________________________________________________________________
|                                                                         |
| OSCILLATOR    SQUARE-CUTOFF         PREVIEW                             |
|     _             normal               on                         OK    |
|    /|\             _                  _                         _____  |
|    \_/              \_               \_                       |     | |
|                     |\|              |\|                      |_____| |
|                     |__|              |__|                              |
|                                                                         |
|                     half                off                             |
|_________________________________________________________________________|
                                                                            |
```

(fig 4l: the pref-screen)

The potmeter OSCILLATOR changes the shape of both the sawtooth and square.
Directing to the left it is become very straight waveforms, and to the
maximum right it become more electric, more natural waves. I prefer the
last one (max to right) coz it sounds fatter, and also the original 303’s
shapes are like these.

The SQUARE-CUTOFF switch determines if the cuttof frequency is the half
sensitive for the square, so it means the sound of the square will be twice
as sprankling, the cutoff is twice as high. Sorry it’s very confusing, but
HALF means DOUBLE! But most of the times you will keep it NORMAL, so set it
to normal (default is also normal). The only use of it is if you want very
bright electro-like sounds.

The preview-option selects if you want the preview ON or OFF. OFF means no
preview while rendering, but only a status-screen of how much is rendered.
If you select Preview ON, you see some info and the just rendered period
waveform, so this is very handy to check what you are doiing and what’s
going on.

OK will let you go back to the main-screen. If you have made wrong choices
in the prefs-screen you can get them back with the UNDO-button on the
main-screen (fig 4m).

```
                                  +----+  +----+ +---+ +----+ +----+ +------+
 PREFS      PLUG-IN               |QUIT|  |UNDO| |NEW| |HELP| |VIEW| |SCRIPT|
+-----+     +-----+               +----+  +----+ +---+ +----+ +----+ +------+
|     |     |     |                          _____________
+-----+     +-----+                         |             |
                                            |_____________|
```

(fig 4m: middle of the main-screen’s buttons)

```
The PLUG-IN button. Clicking on this button gives you a list of plug-ins
available for you to select and use it. A plug-in is a little program that
can be 'plugged' to the main-program. The power of plug-ins is that a
program can be updated by new plug-ins. So a program can by time be
expanded or some plug-ins can be updated. For now, probably only one lug-in
is available. Read the plug-in-helpfiles for instructions about them. The
settings that you can change in the plug-ins, are saved with the LOAD &
SAVE PARAMETERS (see fig 4j).

QUIT - means you leave the program. It asks 1 time if you are sure, and
       then it really quits if you confirm. All settings changed but not
       saved (including rendered but not saved samples) will be lost.

UNDO - undoes the last action

NEW  - sets all the parameters to the default. READ THIS -> if you start
       the 303emulator the settings of the parameters are NOT the default-
       ones! So if you click on NEW, you'll get other settings! This is
       because it now is possible to make a good example for the persons
       who begin for the first time with the program and get a good idea of
       the sound if they press render. Altough maybe it's more confusing to
       understand this story (sorrrry!!!)

HELP - Get this help-file

VIEW - At this moment it doen't anything (anymore, the viewer has been
       dropped)

SCRIPT - Wow! When you select this, you'll get a file-requester where you
         can select. Once pressed OK, it loads and executes it. With this
         script it is possible to generate a whole bunch of samples with
         all different parameters and if you want there can be made a loop
         of all thos samples to one big sample. To read more about this
         script-stuff, click on
                Me
```

1.8 Distribution

```
                                You can get the latest version from :

                  -- APHRODITE 31+ (0)78-6773050 --
                 -- JOLLY WHQ -- 24H -- Sys: Cloak --

              Or wait for the full releases on Aminet(c)



   For other info (printed manual, updates, regristration etc) call :

         --  Jeroen Schellekens, phone 31+ (0)70-3459440
                                 email riff_303@hotmail.com


                Address
```

1.9 Author

```
      Name : Jeroen Schellekens

 Telephone : 31+ (0)70-3459440

     Email : riff_303@hotmail.com

   Address : Paviljoensgracht 60
             2512 BR DEN HAAG
             The Netherlands

 Comments and suggestions are welcome.

 The Code                By Jeroen Schellekens  --     RiFF/JOllY
 The Guide&Installer     By Marinus Kuivenhoven --   aNAKIN/JOllY
```

1.10 History

```
                (Current date is 21 June 1997)


Jan     1996: V0.1  -> Accidently produced a resonant-filter-style sound!

Feb     1996: V1.0  -> First try, released at school (many reactions)

March   1996: V1.1  -> version with better interface, more soundparameters
                       and iff-saver

March   1996: V1.2  -> Last release of this version, with many enhancements,
                       also got in Midi&Recording, a dutch music magazine

April   1996: V1.23 -> Sended to Rinus (JOllY-member, made this guide)

July    1996: V2.0  -> Drastic change of sound AND interface, with new look
                       and sound in the style of the real TB. (hard work!)

Aug-Oct 1996: -------> Bètatesting and debugging period

Novemb  1996: V2.185-> Script-option, new glide-params, better filter,
                       sampleviewer, & many other new options like plug-ins
                       Highpass-filter and Overdrive (new article released
                       about the program in Midi&Recording with screenshots)

Decemb  1996: -------> 'The Party 6'-release

January 1997: V2.2  -> Splatted thos bugs and added new features such as an
                       iff-saver
```

```
May     1997: V2.2a -> Made the interface better, killed the UNDO-bug, made
                       a new Osc/Filter, had new fantasies of what HAS to
                       come in the program (hope it will ever be ready,
                       this horrible program!)
                       Better filerequester, more other reqs, uhm uhm
                       .... uhm ... added a useless button called 'SET'.
                       Will have a function in future...

June    1997: V2.2b -> Tried to release at Aminet (after some dickhead put
                       an old version on it), also got some new parameters.
                       Online help 4 every function.

(Read the
                Futurelist
                !)
```

1.11 Future plans

```
                Plans for future (and they will come out, I promise!):


Apr-Jun 1997: V2.x  -> Busy with an envelope-editor, a tracker-like script-
                       editor and a 16 bit AIFF-saver! This also will be the
                       last release of this version, because the next will
                       be REALTIME!!!!!!
------------------------------------------------------------------------------
Jul-Aug 1997 :    There will be worked on V3 -> THE REALTIME 303Emulator!
                  Yes it's possible, even on a bare A1200 with 2 mb chip
                  and no turbo, to get a realtime 303-emulator. The sound-
                  quality will be like the previous versions (maybe better
                  if I find new tricks), so lotsa better than Rubbaduck on
                  the Pentium-platform.I finally did it with some tips of
                  others to generate the sound in realtime, with some clever
                  tricks of course.

                  What will be possible with the realtime 303Emulator:

                     -program a 303loop in Protracker-style

                     -control in realtime:
                                         the Cutoff-frequency of the filter
                                         the Pitch of the sound
                                         the Resonance of the filter
                                         the Envelope of the filter
                                         the Accent
                                         the Length (Decay) of the sound

                     -1 audio-channel will be used for the 303sound; the
                      other 3 will be free for other samples to program in
                      a Protracker-style.

                     -Midi-syncable with other MIDI-devices
                     -Save the 303track as a sample, for use in other
                      programs.

What is developed yet is:

-A realtime player with only yet the Cutoff-freq,
 Length and Envelope in realtime

-The midi-sync procedure, so you can sync it with another Amiga, that runs
Octamed or another sequencer
------------------------------------------------------------------------------
In whole 1997, a lot of extras could be added like:

            * HOLD CUTOFF_FREQ - wich means that at a certain time the cutoff
              will NOT go down anymore, but keeps stable or just flows down
              very slowly;

            * NOISE WITH RESO - now you can add noisy sounds, and combine
              them with a falling smoothed saw, to make base- and snare-
              sounds;

            * UNSTABLELIZE FILTER - its level determines the noisyness of
              the resonance that can be added with the reso-knob.
              The more unstable, the raw the sound will become. Great for
              dusty bass sound!

            * RESO_FALL - this means that the resonance, say the sines
              added to the sound, will drop in frequency within one
              period of the main-sound, so typical SH-101/MC-202 sounds
              will be possible;

            * A new Fuzz/overdrive function wich will rounden the sound
              after it's clipped to get Wink-style sounds;

            * More effects and plugins...

------------------------------------------------------------------------------
Contact me (Jeroen) (see
                distribution
                ) or JOllY to keep posted, for updates,
new releases and info!
```

1.12 faster

```
    ******************************************
    Quick reference, tips, keyfunctions etc...
    ******************************************

Tip1.
-----
First this: With a 'knob', I mean a rotary-knob or rotary-control, get it?

Tip 2.
------
```

If you want a short description of a knob, button, or parameter, just press help, while pointing the button or parameter. Then, if you are pointing at really something, a small window will appear with some info about the questioned button or parameter.

Tip 3.

All numberkeys and characterkeys represent a piano-like-klavier; when a sample is in memory, thus rendered, you can play the sample by triggering a key. The pitch of the sample depends on wich key you press.

Tip 4.

Mousepointer pointing on a rotary knob:

Mousekey-status	Effect
Nothing pressed	Reports current value in display (on center of screen)
Left button	Decreases value with factor 1x,5x or 20x depending on roughness, set with [SPACE] bar
Right button	Increases value with factor as above
Both buttons	Enter a new value for the current knob

Tip 5.

Mousepointer clicked on the head of Parameter-box A or B:

```
With right mousekey: Able to swap parameterbox with B
With  left mousekey: Able to copy selected parameterbox over other
```

You see a small rectangle while holding a mousebutton, while you hear a sound. Drag it to the other and loose the mousebutton: A copy or a swap will follow!

Tip 6.

Press Space when you are editing the knobs, for reference. When you are changing the pitch for example, and you want to hear if the pitch you are looking for is set well, just press [SPACE] to test it!

Tip 7.

When rendering, and you want to break the process, press BOTH the mouse-buttons a while, until you hear a bleep (end of rendering) and you get back to the main program. The part that IS rendered, you can hear (so it plays the sample until the rendering has cancelled) by playing the keyboard or click on the HEAR-button.

Keyboard shortcuts in program.

```
F1=Render single sample/sequence (Renders selected Param-box (A or B))
F2=Save rendered sample (saves the sample generated in single mode)
F3=Load parameters (loads new settings into the knobs)
F4=Save parameters (saves all your changed settings)
F5=Load and execute script (this is an ASCII-text)
F6=Copy first set of params (A) to last set (B) of params
F7=Copy last set (B) to first set (A)
```

```
F8=Swap first set (A) with last (B)
F9=Prefs-screen
F10=Undo last changed knob-setting

[SPACE]= A preview of the real sample. Very handy for testing!
[TAB]=Switch between Parambox (A) and (B) (determines wich settings 2 use)
[CTRL]=Set rotary-knob-step level (1x=fine/5x=medium/20x=rough)
[HELP]=This helpscreen
[DEL]=Toggle On/Off the graphic of the rendering sample while it renders
[ESCAPE]=Quit
[LEFT SHIFT] while pointing on a knob=Enter a new value for current knob
```

1.13 New plan!

I'm working on a realtime version, wich surely will run on a bare A1200,
and maybe on an A500. So you do not need a fat PC-200 Mhz shit-thing to
get realtime stuff...

For the exact schedule, read the
Futurelist
.

1.14 Concurration

(April '97)

Of course, things change in time. So now, a year later, other people also
have tried to emulate that nasty beast, by writing a piece of software.

I've tested the following 303emulators (but there are more on the market):

Rubberduck (PC, realtime)
338 Rebirth (PC, realtime)
303MU (AMIGA, not realtime(..mmm..))

(...and I have to say that the PC-programs are impressive, but... they're
NOT the real thing, in the way of the sound, and the way of programming.)

Rubberduck is a nice very simple program where you can write simple
tracker-style patterns and play them in realtime, while shifting some
sliders like cutoff etc. It is very easy, and it sounds good, but it misses
a kick, a real analog feeling. Maybe a bit coz of the windows-environment,
but also the spectrum of the sound isn't high. It seems as if it is pre-
rendered, or limited as if it is a digital filter, and it also sounds a bit
8 bitty. So it's very worth a look, maybe it can stand on itself, but it's
not a serious acidmachine-program.

338 Rebirth is another realtime proggie, well a big piece of

software wich is very polished and well designed. The interface consists 2, yep 2 303's above each other and an 808 beneath them. You can program them just as the original TB's and TR, and you can control all the knobs on the virtual machines in realtime. So it must have the feeling of real machines. But it doesn't. I haven't played very long with it (demoversion), so maybe I didn't get the picture right. But the programming -it all must be done with the mouse- and meanwhile rotating the knobs and sliders, you can't do that with one mouse! Despite that, the sound is very very realistic, I have to say. Only again not exactly the real thing. It's a bit too smooth, polished. There are very many options (coz of the demoversion I couldn't test them) like MIDI-controllers, save-track-as-sample and MIDI-syncing stuff. It also costs a lot, 450 guilders or so. But it's a good programmed package. Still it's nodda real thing. (funny to write down bad english for coolness, while making real language-errors without realising it, whihihi!)

303MU is a script-driven program that generates a track, a list of notes based on a textfile. The result sounds very apart, not really 303ish but still acid. But it's fucking shit to program in this way, without an interface. It's better than nothing...

(There are more 303emulators made wich I haven't heard, so maybe I missed the real clone.)

So where stands my program. I think it can survive these programs. Because it is a bit a program of it's own (now I'm talking about my 303emu v2). It maybe does not sound exactly like the real thing, tough it has it's own heavy, resonating acid-type sound, wich can compete very well. This is not my very own opinion. My sound has the same imperfect, impredictable sound as the 303. Besides that, the program its interface makes it all a bit different too. With my new plans (hardware filter!!,external analog rotary knobs!!) the amiga itself becomes a strange soundmachine, a mystic piece of technonology, what, after all, is the meaning of it.

1.15 Changes

The changes in 303emulator V2.2b (13 june 1997) are:

- Finally, finally, a good oscillator, with a better filter. I've been working on this since the beginning of the 303emu.

- Online help, now also available for every function. If you want to know what a certain button or parameter represents, just point at it with your mousepointer and press [HELP]. Then a small window will apear with some info about that button.

- Preview the sound that you want to make: Press [SPACE] to pre-'hear' what settings you made, and a sound will be played with those settings in realtime! So, it's a kind of OpenGL for sound, mmmm... But, it's very handy to test if you've made the settings right, before rendering! You can also save this preview, by clicking on the 'Save'-button above the black little screen.

- New parameters to use, like Reso-fall, to give a more metallic effect. And Hold-CutOff, that will stop decreasing the cutoff frequency of the filter, so you can make kind of MC-202 type sounds.

1.16 303scripts

```
---------------------------------------------------------------------------
Short guide about the scripts that can be loaded in the 303emu, and how
you can make them yourself -> 26-12-96 by Jeroen Schellekens.
---------------------------------------------------------------------------

What’s tha meaning of it?
-------------------------
With 303scripts you can make a list of sounds (samples) where each sound
can have it’s own settings, and there’s no limit of how many sounds u want
to render. Each sound that will be rendered can have all available
parameters used in the 303emulator, but it isn’t nescessery to program each
parameter its value for a sound, so you can set the first sound all its
parameters to certain values, and the second sound can be modified by only
one parameter or two, while unchanged parameters, set with the first sound,
will be used. The scripts can be made and edited with a text-editor.


The commands.
---------------
The following commands are available now (in future more 2 come!):
(All commands must be lowercase!!!!!!!!)

Sound-parameters (sound params always have a value stuck to it!):

w : waveform. Values: 0=sawtooth, 1=square.
t : tuning. Values between 20 and 719 are possible.
c : cutoff frequency. value between 1 and 256 are possible.
r : resonance. Values between 0.00 and 1.00 are possible (floats).
e : envelope. Values between 0.00 and 1.00 (floats).
d : decay. Values between 1 and 200.
a : accent. Values between 1.00 and 2.00 are possible (floats).
p : volume level. Values between 0.00 and 4.00 (floats).
o : distortion. Values between 1.00 and 4.00 (floats).
g : glide up or down. Values: 0=down, 1=up.
l : glide level. Values between 0 and 1.2 (floats).
s : glide start-point. Values between 1 and 200.
z : glide end-point. Values between 1 and 200.
h : highpass-level. Values between 0 and 127.
v : overdrive-level. Values between 1 and 10.


Other commands:

the ’-’ sign : means that it is an empty sample. So there will be saved a
              sample that is empty! This is very useful, if you want to make
              a loop of your script-rendered samples (see below). - commands
              have the effect that there is no sample at that point in the
              loop, so at that point it will be quiet or the last sample
              will be played until its end. It’s unmissable to make loops.

end : This means that the script has ended. This command must be set
```

```
      always, or the stuff will crash (I think. Try it out yourself!)

scriptbas 'filename' : sets the basename of the sample that will be
                       rendered (optional, a defaultname will be used, if
                       not set. This defaultname is set in the "def.pref"
                       file you can find in "303emu2:system/", and you can
                       edit that def.pref file in a normal ASCII-editor.)

makeloop bpm___ hz_____ 'filename' : make a loop of the 'til now rendered
                                     samples, and after the word bpm a
                                     value must be stuck, between 25 and
                                     255. Bpm stands for beats per minute
                                     and determines the tempo. Second, the
                                     hz must be en tered followed with the
                                     frequency in Hertz. Then, a filename
                                     must be given. So for example
                                     something like this --+
                                                           |
makeloop bpm140 hz22000 'work:loop1'             <---------+

                                     Also, this command is optional, not
                                     necessery.

The structure of a script.
--------------------------
Next, you see an example script:


'303-example-script
'© 1996 Jeroen Schellekens

scriptbas 'work:303emu2/sample.'

t230  c16  r0.996  e0.06  d4  a1  p1.4   o1  g0  l0    s1  z1  w0
t220  c15
t192  c16
t192  c15
t180  c16                 d10                    l0.15 s9  z14
-
t220  c19                 d4                     l0            w1
t180              e0.1

makeloop bpm125 hz17000 'work:303testloopa'

end



The head of the script begins with a ' sign, followed some text. This '
sign means that it is a remark, so you can write information after it to
remember what it or something else, while it hasn't got any effect or
function for the program. It's just handy for you.

Also, spaces and empty lines have no function or effect, they just make
it more arranged for you.

Then we see a word scriptbas followed by two ' signs with a path and name
```

```
between them. "scriptbas" is the short for script-base-sample-name or
something like that, and it's function is the basename for the sample that
will be saved. After this basename, the number of the sample will be
added, such as "sample.001".

Again, remember that all the commands must be lowercase, not uppercase,
or it will not work!

Then we see (after an empty line) a set of characters, and stuck to them
you see values. The command is the parameter, and the value is ... the
value that belongs to that parameter. So in the current line the first
parameter is t (for tuning) and its value is 230.
You see, the first line with parameters contains many parameters or
commands, but the second one contains only a tuning (t220) and different
cutoff frequency (c15). This means that the first sample uses the values of
the first line, and the second one uses the the two commands its values,
and the other parameters will not change, so will the same in the second
sample.
Also the following order of the commands isn't important.

The - means that there will be an empty place or sample.

makeloop makes a loop of all samples into one new sample. Now this is where
the - sign comes in. Its function is that on the 6th place, no sound will
be heard and that the 5th sample will play to its end. The tempo is
controlled by bpm, in this case 125 beats per minute. The hz-command
determines the playbackrate, and the higher the value after it, the higher
the quality will be. Last, the name of the sample must be given, between to
' signs.

Then we see end, and this command must be in every script. If not, the
program will not know that it is ready, and will loop until you reset your
computer.

Hope I've given enough explanation 'bout the script-language.
---------------------------------------------------------------------------
```

1.17 Projects

```
                Amiga-Analog-Synth.
-------------------
I am now trying to control a home-build (by my uncle) 3-pole 18 dB filter
(with Lowpass,Bandpass,Highpass,Cutoff and Resonance(Q)-knobs) by the
Amiga, where the envelope of this filter are controlled by the Amiga.
Meanwhile, up to 4 Oscillators can be individually played with different
waveforms, pitches, and even PWM, Osc-sync and Rig-mod will be pretty easy.
These waveforms will be all lead through the 3-pole filter.
All the shit will be in the form of a module wich will be controlled by
MIDI.
I am saying, it's really not ready yet, and maybe it will not succeed to
control the filter. Any sugestions (about how to convert digital signals to
analog resistor-values, MIDI-control etc) are very very welcome. What also
would be cool is to send data via a port (serial or parallel) to a C64, to
controll the whole SID-chip of that old beast! But I haven't got the
knowledge and experience with that, so people: HELP ME
```

```
                  ->Address
```